10.1-numpy

December 19, 2024

Numpy NumPy is a fundamental library for scientific computing in Python. It provides support for arrays and matrices, along with a collection of mathematical functions to operate on these data structures. In this lesson, we will cover the basics of NumPy, focusing on arrays and vectorized operations.

[1]:
```
!pip install numpy
```

```
Requirement already satisfied: numpy in
/home/test/anaconda3/envs/DNN/lib/python3.12/site-packages (1.26.4)
```

[3]:
```
import numpy as np

## create array using numpy
##create a 1D array
arr1=np.array([1,2,3,4,5])
print(arr1)
```

```
[1 2 3 4 5]
```

[4]:
```
print(type(arr1))
```

```
<class 'numpy.ndarray'>
```

[5]:
```
print(arr1.shape)
```

```
(5,)
```

[9]:
```
## 1 d array
arr2=np.array([1,2,3,4,5])
arr2.reshape(1,5)  ##1 row and 5 columns
```

[9]:
```
array([[1, 2, 3, 4, 5]])
```

[11]:
```
arr2=np.array([[1,2,3,4,5]])
arr2.shape
```

[11]:
```
(1, 5)
```

[6]:
```
## 2d array
arr2=np.array([[1,2,3,4,5],[2,3,4,5,6]])
```

```
print(arr2)
```

```
[[1 2 3 4 5]
 [2 3 4 5 6]]
```

[7]:
```
print(arr2.shape)
```

```
(2, 5)
```

[14]:
```
np.arange(0,10,2).reshape(5,1)
```

[14]:
```
array([[0],
       [2],
       [4],
       [6],
       [8]])
```

[15]:
```
np.ones((3,4))
```

[15]:
```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

[16]:
```
## identity matrix
np.eye(3)
```

[16]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[8]:
```
## Attributes of Numpy Array
arr = np.array([[1, 2, 3], [4, 5, 6]])

print("Array:\n", arr)
```

```
Array:
 [[1 2 3]
 [4 5 6]]
```

[14]:
```
print("Shape:", arr.shape)  # Output: (2, 3)
```

```
Shape: (2, 3)
```

[10]:
```
print("Number of dimensions:", arr.ndim)  # Output: 2
```

```
Number of dimensions: 2
```

[11]:
```
print("Size (number of elements):", arr.size)  # Output: 6
```

```
Size (number of elements): 6
```

[12]:
```
print("Data type:", arr.dtype)  # Output: int32 (may vary based on platform)
```

```
Data type: int64
```

[13]:
```
print("Item size (in bytes):", arr.itemsize)  # Output: 8 (may vary based on␣
↪platform)
```

```
Item size (in bytes): 8
```

[15]:
```
### Numpy Vectorized Operation
arr1=np.array([1,2,3,4,5])
arr2=np.array([10,20,30,40,50])

### Element Wise addition
print("Addition:", arr1+arr2)
```

```
Addition: [11 22 33 44 55]
```

[16]:
```
## Element Wise Substraction
print("Substraction:", arr1-arr2)
```

```
Substraction: [ -9 -18 -27 -36 -45]
```

[17]:
```
# Element-wise multiplication
print("Multiplication:", arr1 * arr2)
```

```
Multiplication: [ 10  40  90 160 250]
```

[18]:
```
# Element-wise division
print("Division:", arr1 / arr2)
```

```
Division: [0.1 0.1 0.1 0.1 0.1]
```

[19]:
```
## Universal Function
arr=np.array([2,3,4,5,6])
## square root
print(np.sqrt(arr))
```

```
[1.41421356 1.73205081 2.         2.23606798 2.44948974]
```

[20]:
```
## Exponential
print(np.exp(arr))
```

```
[  7.3890561   20.08553692  54.59815003 148.4131591  403.42879349]
```

[21]:
```
## Sine
print(np.sin(arr))
```

```
[ 0.90929743  0.14112001 -0.7568025  -0.95892427 -0.2794155 ]
```

[22]:
```
## natural log
print(np.log(arr))
```

```
[0.69314718 1.09861229 1.38629436 1.60943791 1.79175947]
```

[23]:
```
## array slicing and Indexing

arr=np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])
print("Array : \n", arr)
```

```
Array :
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

[34]:
```
print(arr[1:,1:3])
```

```
[[ 6  7]
 [10 11]]
```

[25]:
```
print(arr[0][0])
```

```
1
```

[26]:
```
print(arr[0:2,2:])
```

```
[[3 4]
 [7 8]]
```

[29]:
```
arr[1:,2:]
```

[29]:
```
array([[ 7,  8],
       [11, 12]])
```

[36]:
```
## Modify array elements
arr[0,0]=100
print(arr)
```

```
[[100   2   3   4]
 [  5   6   7   8]
 [  9  10  11  12]]
```

[37]:
```
arr[1:]=100
print(arr)
```

```
[[100   2   3   4]
 [100 100 100 100]
 [100 100 100 100]]
```

[38]:
```
### statistical concepts--Normalization
##to have a mean of 0 and standard deviation of 1
data = np.array([1, 2, 3, 4, 5])

# Calculate the mean and standard deviation
```

```
mean = np.mean(data)
std_dev = np.std(data)

# Normalize the data
normalized_data = (data - mean) / std_dev
print("Normalized data:", normalized_data)
```

```
Normalized data: [-1.41421356 -0.70710678  0.          0.70710678  1.41421356]
```

[27]:
```
data = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

# Mean
mean = np.mean(data)
print("Mean:", mean)
```

```
Mean: 5.5
```

[28]:
```
# Median
median = np.median(data)
print("Median:", median)
```

```
Median: 5.5
```

[29]:
```
# Standard deviation
std_dev = np.std(data)
print("Standard Deviation:", std_dev)
```

```
Standard Deviation: 2.8722813232690143
```

[30]:
```
# Variance
variance = np.var(data)
print("Variance:", variance)
```

```
Variance: 8.25
```

[31]:
```
## Logical operation
data=np.array([1,2,3,4,5,6,7,8,9,10])

data[(data>=5) & (data<=8)]
```

[31]:
```
array([5, 6, 7, 8])
```